A Novel Crossover Operator in Evolutionary Algorithm for Logic Circuit Design

Guo-liang He[1,2], Yuan-xiang Li[1], and Zhongzhi Shi[2]

[1] State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China
[2] Key Laboratory of Intelligent Information Process, Institute of Computing Technology, The Chinese Academy of Sciences, Beijing, China
glhe@whu.edu.cn, yxli@whu.edu.cn, shizz@ics.ict.ac.cn

Abstract. In recent years the evolution of digital circuits has been intensively studied. This paper proposes an elitist pool evolutionary algorithm based on novel approach in order to improve evolutionary design of logic circuits in efficiency and capability of optimization. In the process of evolution, a novel sub-circuit crossover strategy can improve the local optimization by inheriting the better parts of two parental circuits, and an adaptive mutation strategy based on importance of gene-position can maintain the diversity of a population. Experiments show that the proposed method is able to design logic circuits efficiently.

Keywords: Evolvable hardware, evolutionary algorithm, logic circuit.

1 Introduction

In 1990s, Hugo de Gairs advanced a new hardware design method, *evolvable hardware (EHW)*, based on the method of evolutionary computation (EC) and the reconfigurable characteristics of FPGA. By simulating the evolution process in nature, EHW is used to design physical circuits, especially electronic circuits. As a new multi-discipline research field involved in computer science, electronic engineering and biology, EHW provides a feasible method for automation design and improving intelligence of hardware systems, such as adaptive filter, intelligent controller, intelligent antenna systems, etc [1-3].

Currently, the research of EHW is mainly on the auto-design of electronic and analog circuits. And a great number of algorithms for EHW have been proposed with encouraging results. For instance, Koza designed a single uniform approach using genetic programming for the automatic synthesis of both the topology and sizing of different prototypical analog circuits, including a lowpass filter, a source identification circuit, an amplifier, etc [4]. Emanuele Stomeo et al. proposed generalized disjunction decomposition (GDD) [5] based on rewriting the truth table in such a way that the inputs needed to describe the system were decomposed in a lower-level circuit and a multiplexer. It is beneficial for solving more complex logic circuits, not obtaining optimized logic circuits. Meanwhile, other intelligent approaches such as multi-objective

Z. Cai et al. (Eds.): ISICA 2009, CCIS 51, pp. 110–117, 2009.

approaches [6-7], variable length chromosome GA [8], and incremental development evolution [9-10], were also proposed for automated design of logic circuit.

In this paper, we investigated a novel elitist pool evolutionary algorithm (EPEA) for designing logic circuits. It employs evolutionary strategies to quicken the local optimization and maintain the diversity of a population in terms of the characteristics of logic circuits. In section 2, we briefly describe the encoding method of logic circuits. Section 3 presents a novel elitist pool evolutionary algorithm, including a novel evolutionary operator, an adaptive mutation strategy and an evaluation strategy. Section 4 gives some examples to analyze the performance of the proposed approach. Finally, we discuss conclusive remarks in Section 5.

2 Representation of Individual

We use a matrix structure to represent digital circuits as shown in Fig. 1. The inputs of each cell are the original inputs or other cells' outputs, and the outputs of last column are the circuit's outputs. In order to represent combinational logic circuits fully, definitions of the function types of cells and the connectivity among cells are required for the matrix. In this matrix, each cell represents a logic gate, and some function types of gates are shown in Table 1. For the connectivity, we define that only previous cells connect to next cells to avoid a circle. The outputs of the cells of the last column in the matrix are the outputs of the encoded circuit respectively and the remaining cells of the last column are redundant.

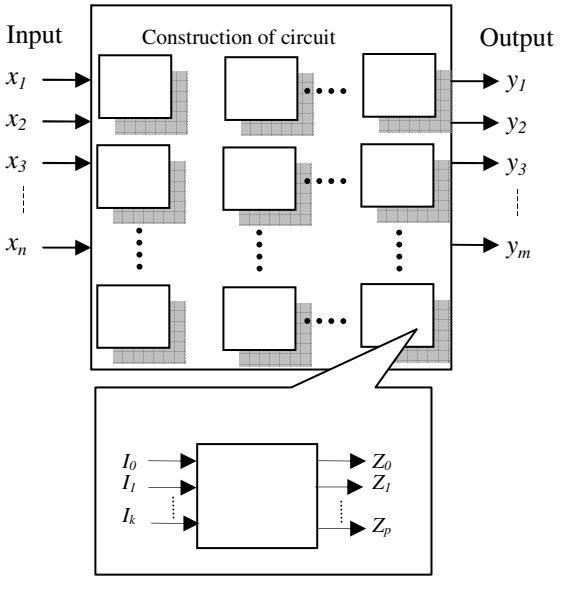

Fig. 1. The matrix encoding representation for digital circuits

Table 1. Function types of basal logic gates

	Function type		Function type
0	A*	3	A + B
1	!A	4	$A \oplus B$
2	A • B	5	A • !C + B • C

* This function type means the first input of a cell is directly connected to its output, without employing any logic operators.

3 Elitist Pool Evolutionary Algorithm

In general, local search optimization and the population diversity are two basic principles in evolutionary algorithm. Based on the principles and characteristics of logic circuits, an elitist pool evolutionary algorithm with a new crossover operator and an adaptive mutation strategy is introduced as follows.

3.1 Sub-circuit Crossover Operator

The evolution design of logic circuits is a multi-objective optimization problem when every output of the circuit is considered as an objective which is a sub-circuit containing some cells and their connections. Therefore, the structure of a circuit can be formed by the fusion of all sub-circuits of outputs. Based on the local optimization, a sub-circuit of a new individual can be selected from the better parental sub-circuits corresponding to the same objective in the process of the crossover. In order to obtain correct circuits with the minimized number of logic gates, sharing cells must be widely used while all sub-circuits are fused into a whole circuit. Therefore, some cells of these selected parental sub-circuits should be implemented by some replacing approaches to form a new integrated circuit instead of the collection of all sub-circuits.

In this paper, the replacing method is designed in terms of outputs location. For example, the fitness value of each output of two parents with 3 inputs and 2 outputs is 7/8, 4/8 and 5/8, 6/8 respectively shown in Fig. 2. Each cell is labeled to which sub-circuit it belongs. For instance, symbol “1” or “2” respectively denotes that it belongs to the first or the second sub-circuit, and symbol “-1” denotes this cell does not belong to any sub-circuits but a redundant cell. A cell noted “1,2” means this cell is a shared logic gate of both the first and the second sub-circuit. The sub-circuit of the first output is firstly selected from the better corresponding sub-circuit of parent I, and the sub-circuit of the second output is selected from parent II. At the same time, some cells of the first sub-circuit should be replaced when these cells are conflicted with the second sub-circuit in the same positions of this matrix.

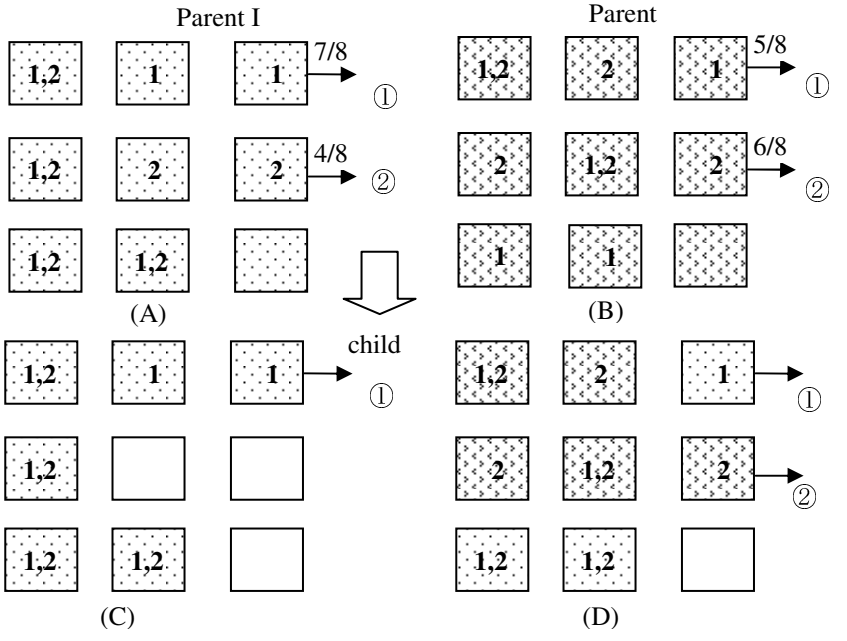

Fig. 2. The process of sub-circuit crossover from parents: (A) and (B) are the parents; (C) the first sub-circuit of a new individual from parent 1, which is better than parent 2; (D) an new integrated individual is produced after the fusion of the three sub-circuits from parents.

3.2 Adaptive Mutation Strategy

According to the importance of gene-position in the chromosome, the important genes are mutated with a lower rate to enhance the convergence of the algorithm, and the less important genes are mutated with a high rate to maintain the diversity of population. In terms of the encoding scheme used here, each cell of the rectangular array corresponds to a section of genes, and its sharing degree determines its importance. The higher the sharing degree is, the lower the rate is. At the same time, considering the function of each circuit, the mutation rate of a cell will be increased if its corresponding function of a sub-circuit including this cell does not completely match the truth table; otherwise, it will be decreased.

This is essential for the evolution of circuits. It has a bigger chance for the better part of the circuit to be reserved while the worse part may be possible to be changed drastically.

3.3 Framework of Elitist Pool Evolutionary Algorithm

The main process of elitist pool evolutionary algorithm is as follow:

Step1: Initialize a host population A and its scale is N. And evaluate each individual of this population.

Step2: Create M individuals to obtain an elitist pool B by mutating the best individual in the host population A, where N>>M.

Step3: The sub-circuit crossover operator is performed to create a new individual, where a parent is randomly selected from the elitist pool and another is selected from the host population respectively. Then mutate this new individual. Repeating this step N times until obtaining a new host population A.

Step4: Evaluate the new host population and reserve the best individual.

Step5: Go to step2 to continue this algorithm until the ending condition is satisfied; otherwise, end this algorithm.

3.4 Evaluation

In the process of circuit evolution, evaluation criterion is an important part, which reflects the performance of an evolved circuit and guides the next evolution. Firstly, it is vital for evolved circuits to have correct functions or behaviors. For the function of digital circuit, the evaluation model is realized by using testing data set. Secondly, a functionally correct circuit with fewer logic gates is usually preferable, which occupies fewer areas. And it is also natural to measure a circuit's efficiency of resource usage by the number of gates it uses.

According to the proposed ideas, in this paper the fitness function $F(x_i)$ evaluating an encoded circuit x_i is defined as follows:

$$F_1(x_i) = H(x_i) + w*V(x_i)$$

Here, $H(x_i)$ is defined as the matching degree by comparing the function of all output with the truth table; $V(x_i)$ is the number of cells whose function type are *wire*s in the circuit x_i and the redundant cells in the matrix; w is a weight coefficient, which changes in the different phases.

4 Experiment

Three case studies including 2-bit adder, 2-bit multiplier and 2-bit comparator are carried out using EPEA and its parameters are as follow: the population size N=100, maximum of number of generations T=20000; the mutation rate is initialized as 0.08 and the crossover rate is 0.8. Here we use gate-level and function-level evolution to design these arithmetic circuits. In order to simplify the function type, in this paper the function types of basic gates are shown in Table 1, and the function blocks include these basic gates and 2-multiplexers.

For the 2-bit adder, by function-level method, EPEA could obtain an optimized 2-bit adder circuit at 100 percent by evolving about 2800 generations when the size of the encoding matrix is 3×3, and its circuit is shown in Fig. 3. However, it could not get a correct circuit with the same size of the encoding matrix by gate-level method. When the size of the encoding matrix increases to 4×4, EPEA could get an optimized 2-bit adder circuit as follow in Fig.4.

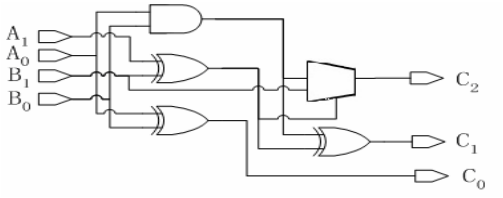

Fig. 3. Circuit design of 2-bit adder with function-level approach

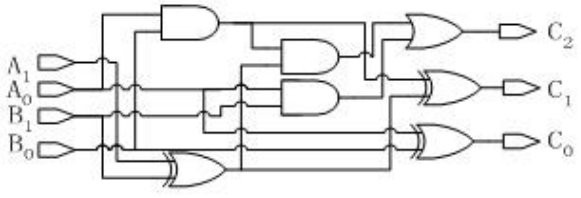

Fig. 4. Circuit design of 2-bit adder with gate-level approach

For the 2-bit multiplier, the size of the encoding matrix is 4×4, EPEA could get the same optimized 2-bit multiplier circuit with 7 gates by gate-level and function-level approaches and one of these optimized circuits is shown in Fig.5.

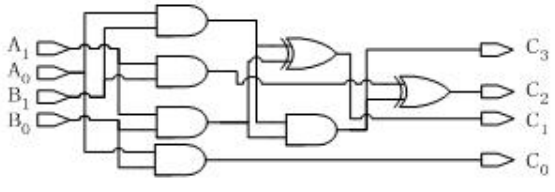

Fig. 5. Circuit design of 2-bit multiplier

For the 2-bit comparator, the size of the encoding matrix is 5×5. By function-level method, EPEA could obtain an optimized 2-bit comparator circuit with 7 gates as shown in Fig. 6 while an optimized 2-bit comparator circuit with 9 gates by gate-level evolution is shown in Fig.7.

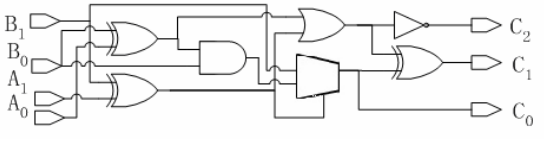

Fig. 6. Circuit design of 2-bit comparator with function-level approach

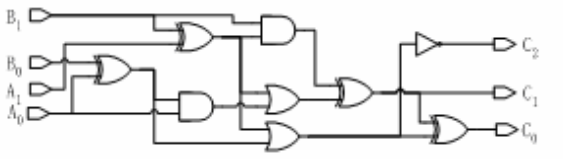

Fig. 7. Circuit design of 2-bit comparator with gate-level approach

On the other hand, the best solutions obtained by NGA [11], MGA [12], KM [13], QM [14-15] and EPEA for the forementioned circuits are also compared in Table 2 (The symbol "Size" represents the size of the matrix, and "Num" denotes the number of gates in the optimized circuit). From these results we see that EPEA can similar optimized circuits by representing them with smaller size of the matrix, which is adaptive for designing evolvable hardware.

Table 2. Comparison of the best solutions for combinational logic circuits

Method	2×2 multiplier		2-bit comparator		2-bit adder	
	Size	Num	Size	Num	Size	Num
KM	—	8	—	19	—	12
QM	—	12	—	13	—	—
NGA	5×5	9	6×7	12	5×5	7
MGA	5×5	7	6×7	9	5×5	7
EPEA	4×4	7	5×5	9	4×4	7

5 Conclusion

In this paper, a novel elitist pool evolutionary algorithm is advanced to design logic circuits automatically and efficiently according to the characteristics of logic circuits. Experiments on auto-design of some combinational logic circuits show the proposed evolutionary strategies can be extraordinarily encouraging in efficiency and capability of optimization, and some of evolved circuits can be used as basic macro-blocks to design larger logic circuits automatically.

Acknowledgments

This project was supported by National Natural Science Foundation of China (under grants No. 60773009) and National High-Tech Research and Development of China (No. 2007AA01Z290).

References

1. Linden, D.S.: Optimizing signal strength in-situ using an evolvable antenna system. In: 2002 NASA/DoD Conference on Evolvable Hardware, pp. 147–151 (2002)
2. Vinger, K.A., Torresen, J.: Implementing evolution of FIR-filters efficiently in an FPGA. In: 2003 NASA/DoD Conference on Evolvable Hardware, pp. 26–29 (2003)

3. Glette, K., Torresen, J., Gruber, T., Sick, B., Kaufmann, P., Platzner, M.: Comparing Evolvable Hardware to Conventional Classifiers for Electromyographic Prosthetic Hand Control. In: NASA/ESA conference on adaptive hardware and systems, Noordwijk, the Netherlands, June 2008, pp. 32–39 (2008)
4. Koza, J.R., Bennett, F.H., Andre, D., Keane, M.A., Dunlap, F.: Automated Synthesis of Analog Electrical Circuits by Means of Genetic Programming. IEEE Transactions on Evolutionary Computation 1(2), 109–128 (1997)
5. Stomeo, E., Kalganova, T., Lambert, C.: Generalized Disjunction Decomposition for Evolvable Hardware. IEEE Transactions on, Systems, Man and Cybernetics (Part B) 36(5), 1024–1043 (2006)
6. Ishida, Y., Nosato, H., Takahashi, E., Murakawa, M., Kajitani, I., Furuya, T., Higuchi, T.: Proposal for LDPC Code Design System Using Multi-Objective Optimization and FPGA-Based Emulation. In: The 8th internalional Conference on Evolvable Systems: From Biology to Hardware, Prague, Czech Republic, September 2008, pp. 237–248 (2008)
7. Coello, C.A.C., Aguirre, A.H.: Design of combinational logic circuits through an evolutionary multiobjective optimization approach. Ai Edam-Artificial Intelligence for Engineering Design Analysis and Manufacturing, 16(1), 39–53 (2002)
8. Kalganova, T., Miller, J.F.: Circuit layout evolution: An evolvable hardware approach. In: IEE Colloquium (Digest), pp. 11–14 (1999)
9. Miller, J.F., Job, D., Vassilev, V.K.: Principles in the evolutionary design of digital circuits—part II. In: Genetic Programming and Evolvable Machines, vol. 1, pp. 259–288 (2000)
10. Bidlo, M., Vasicek, Z.: Gate-Level Evolutionary Development Using Cellular Automata. In: NASA/ESA conference on adaptive hardware and systems, Noordwijk, the Netherlands, pp 11–18 (June 2008)
11. Coello Coello, C.A., Christiansen, A.D., Aguirre, A.H.: Use of Evolutionary Techniques to Automate the Design of Combinational Circuits. International Journal of Smart Engineering System Design 2, 299–314 (2000)
12. Coello Coello, C.A., Aguirre, A.H.: Design of combinational logic circuits through an evolutionary multiobjective optimization approach. Artificial Intelligence for Engineering, Design, Analysis and Manufacture 16(1), 39–53 (2002)
13. Karnaugh, M.: A Map Method for Synthesis of Combinational Logic Circuits. Transaction of the AIEE, Communications and Electronic 72(I), 593–599 (1953)
14. Quine, W.V.: A Way to Simplify Truth Function. American Mathematical Monthly 62(9), 627–631 (1955)
15. McCluskey, E.J.: Minimization of Boolean Function. Bell Systems Technical Journal 35(5), 1417–1444 (1956)